CS6160 Theory of Computation
Problem Set 5
Department of Computer Science, University of Virginia

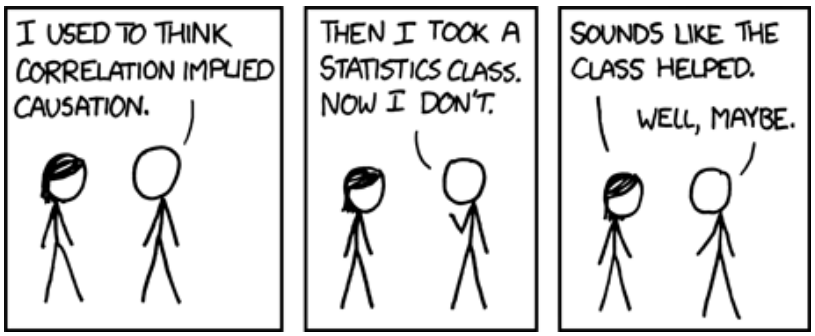
Gabriel Robins

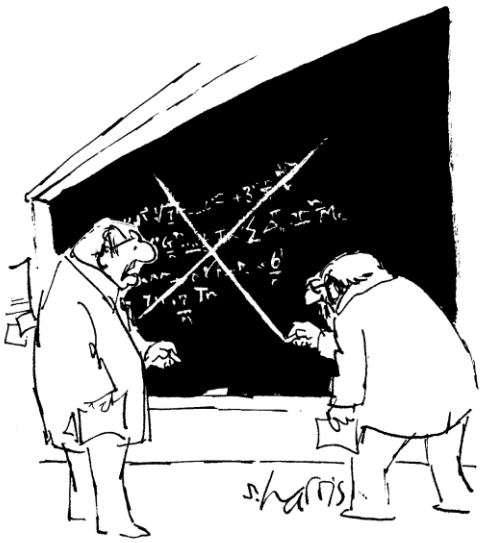
Please start solving these problems immediately, don't procrastinate, and work in study groups. Please prove all your answers; informal arguments are acceptable, but please make them precise / detailed / convincing enough so that they can be easily made rigorous if necessary. To review notation and definitions, please read the "Basic Concepts" summary posted on the class Web site, and also read the corresponding chapters from the Sipser textbook and Polya's "How to Solve It".

Please do not simply copy answers that you do not fully understand; on homeworks and on exams we reserve the right to ask you to explain any of your answers verbally in person (and we have exercised this option in the past). Please familiarize yourself with the UVa Honor Code as well as with the course Cheating Policy summarized on page 3 of the Course Syllabus. To fully understand and master the material of this course typically requires an average effort of at least six to ten hours per week, as well as regular meetings with the TAs and attendance of the weekly problem-solving sessions.

This is not a "due homework", but rather a "pool of problems" meant to calibrate the scope and depth of the knowledge / skills in CS theory that you (eventually) need to have for the course exams, becoming a better problem-solver, be able to think more abstractly, and growing into a more effective computer scientist. You don't necessarily have to completely solve every last question in this problem set (although it would be great if you did!). Rather, please solve as many of these problems as you can, and use this problem set as a resource to improve your problem-solving skills, hone your abstract thinking, and to find out what topics you need to further focus on and learn more deeply. Recall that most of the midterm and final exam questions in this course will come from these problem sets, so your best strategy of studying for the exams in this course is to solve (including in study groups) as many of these problems as possible, and the sooner the better!

Advice: Please try to solve the easier problems first (where the meta-problem here is to figure out which are the easier ones ☺). Don't spend too long on any single problem without also attempting (in parallel) to solve other problems as well. This way, solutions to the easier problems (at least easier for you) will reveal themselves much sooner (think about this as a "hedging strategy" or "dovetailing strategy").





"That's *it*? That's peer review?"

1. The following problems are from [Sipser, Second Edition]:

   Pages 128-132: 2.4, 2.5, 2.6, 2.9, 2.10, 2.13, 2.16, 2.17, 2.20, 2.21, 2.22, 2.24, 2.27, 2.32, 2.33, 2.36, 2.37, 2.40, 2.41, 2.42, 2.43, 2.44, 2.45

2. Does every context-free language have a proper context-free subset?
   Does every context-free language have a proper context-free superset?

3. Is every subset of a context-free language necessarily context-free?
   Is every superset of a context-free language necessarily non-context-free?

4. Is a countably-infinite union of context-free languages necessarily context-free?
   Is a countably-infinite intersection of context-free languages necessarily context-free?

5. What is the infinite union of all the context-free languages?
   What is the infinite intersection of all the context-free languages?

6. Let YESNO(L)=$\{xy \mid x \in L \text{ and } y \notin L, x,y \in \Sigma^*\}$. Does YESNO preserve context-freeness?

7. Let PALI(L)=$\{w \mid w \in L \text{ and } w^R \in L\}$. Does PALI preserve context-freeness?

8. Given the alphabet $\Sigma=\{a,b,(,),+,*,\emptyset, \varepsilon\}$ construct a context-free grammar that generates all strings in Σ^* that correspond to regular expressions over {a,b}.

9. Construct the smallest possible (in terms of the number of non-terminals and/or production rules) context-free grammar that generates all well-formed parenthesis.

10. Construct a (small) context-free grammar that generates all well-formed nestings of parenthesis ( ) and brackets [ ].

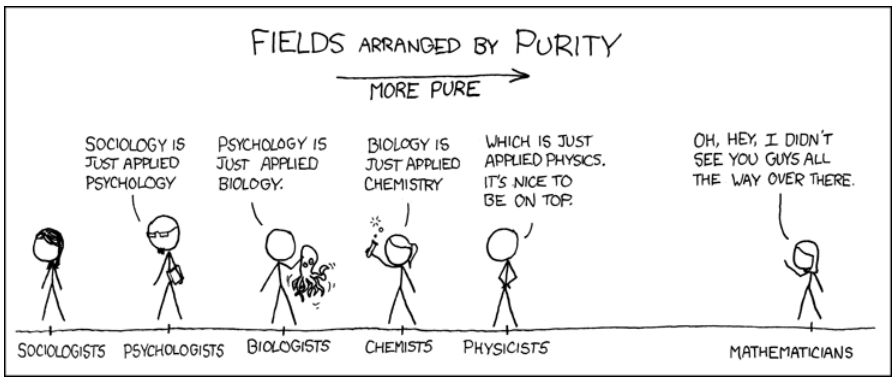
11. Characterize as precisely as you can the class of languages accepted by deterministic pushdown automata with two stacks.

12. Characterize as precisely as you can the class of languages accepted by deterministic pushdown automata with a single "counter" (i.e., stack with only a single-letter stack alphabet).

13. Characterize as precisely as you can the class of languages accepted by deterministic pushdown automata with two "counters" (i.e., two stacks with only a single-letter stack alphabet).

14. Does there exist a context-free grammar for $\{0^i1^j \mid 1 \le i \le j \le 2i\}$?

15. Does there exist a context-free grammar for $\{0^i1^j \mid 0 \le i \le j \le 1.5i\}$?



16. Let $L=\{0^n1^n \mid n\geq 0\}$. Is $\overline{L}$ (i.e. the complement of L) a context-free language?

17. Let $L=\{0^i1^j \mid i \neq j\}$. Is L a context-free language?

18. Is $\{w\in\{a,b\}^* \mid$ w contains an equal number of a's and b's$\}$ a context-free language?

19. We define the SHUFFLE of two strings $v,w \in \Sigma^*$ as:

$$\text{SHUFFLE}(v,w) = \{v_1w_1v_2w_2...v_kw_k \mid v=v_1v_2...v_k,\ w=w_1w_2...w_k,\ \text{and for some } k \geq 1,\ v_i,w_i \in \Sigma^*,\ 1 \leq i \leq k\}$$

For example, 212**ab**1**baa**2**b**22 $\in$ SHUFFLE(**abbaab**,2121222)

Extend the definition of SHUFFLE to two languages $L_1,L_2 \subseteq \Sigma^*$ as follows:

$$\text{SHUFFLE}(L_1,L_2) = \{w \mid w_1\in L_1,\ w_2\in L_2,\ w\in \text{SHUFFLE}(w_1,w_2)\}$$

a) Is the SHUFFLE of two context-free languages necessarily context-free?

b) Is the SHUFFLE of a context-free language with a regular language necessarily context-free?

20. Is $\{v\$w \mid v,w \in \{a,b\}^*, v\neq w\}$ a context-free language?

21. Is $\{vw \mid v,w \in \{a,b\}^*, v\neq w\}$ a context-free language?

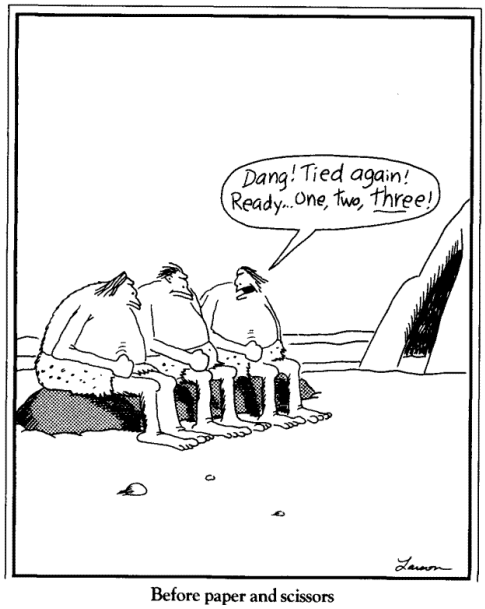
22. Is $\{vw \mid v,w \in \{a,b\}^*, v\neq w, |v|=|w|\}$ a context-free language?

23. Determine whether each of the following languages is context-free.

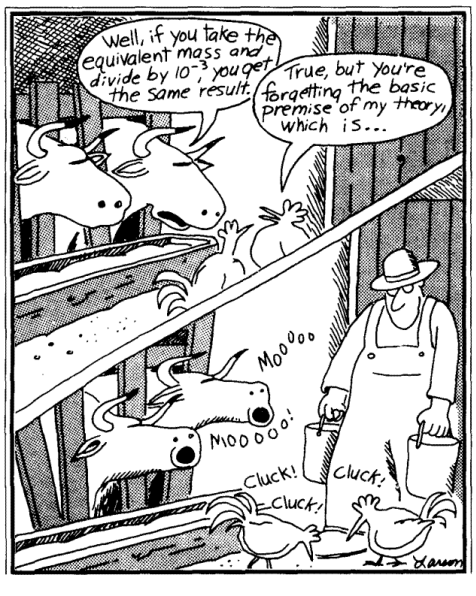
a) $\{a^na^na^n \mid n>0\}$

b) $\{www \mid w \in \{x,y,z\}^*, |w| < 10^{100}\}$

c) $\{vw \mid v,w \in \{a,b\}^*\}$



Before paper and scissors



24. Which of the following modifications / restrictions to PDA's would change the class of languages accepted, relative to "normal" PDA's?

a) The ability to move the read head backwards (as well as forwards) on the input.

b) The ability to write on (as well as read from) the input tape.

c) Having 2 read-heads moving (independently, left-to-right) over the input.

d) Having three stacks instead of one.

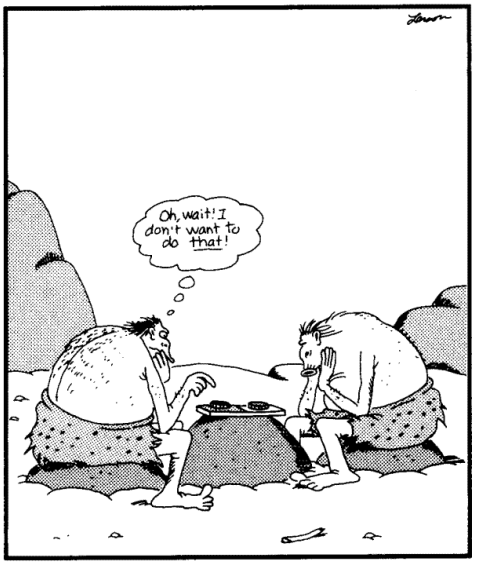
e) Having a stack alphabet of at most two symbols.

f) Having a stack alphabet of one symbol.

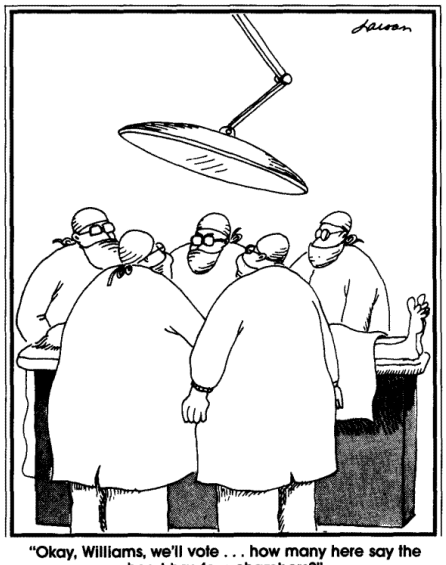
g) Having a FIFO queue instead of a stack (i.e., write-only at the top of the queue, and read-only at the bottom of the queue).

25. Write a program (in your favorite programming language, e.g. C, C++, Java, Python, etc.) that prints itself out (i.e., prints its own source code exactly, character-for-character, including white spaces), without reading any files, input devices, buffers, Web sites, or any other external input sources. (A blank program isn't a solution here – this is an exercise in self-replication.)

26. Write a self-printing program which is as short as possible (i.e. having the least number of characters in its source code). Also give it the ability to contain arbitrarily large "payload" (say an arbitrary subroutine or code section) that gets copied along with the rest of the source code. Name an application of this capability.



Early checkers


"Okay, Williams, we'll vote . . . how many here say the heart has four chambers?"